Glenn Research Center

Setup and Execution of the CFD Simulation

Glenn Research Center

Setup Issues

Setting up the CFD simulation concerns several issues:

- WIND Input Data File
- Time and Space Marching
- Flux Formulation
- Boundary Condition Inputs
- Flowfield Initialization
- Damping Initial Transients

Glenn Research Center

WIND: Input Data File

- Input data file (.dat) is ASCII file.
- File contains descriptive keywords.
- Online documentation lists all keywords.

Example Input Data (*.dat) file:

```
RAE 2822 Airfoil. 2D Transonic Flow.
Mach = 0.729. Alpha = 2.31 deg.
Single zone C-grid 369 x 65.

/Freestream      Mach  p(psi) T(R) AOA  Beta
freestream static 0.729 15.8  460.0 2.31 0.0

downstream pressure freestream zone 1

turbulence model spalart allmaras

dq limiter on

implicit boundary on

cycles 200
iterations per cycle 10 print frequency 10

cfl 5.0

end
```

First 3 lines are for titles.

"/" indicates a comment line .

Freestream keyword sets reference conditions, which are input in units consistent with grid file.

Specifies additional information for the outflow boundary condition.

Specifies choice of turbulence model.

Specifies use of limiter for change in solution (dq) over an iteration.

Specifies use of implicit boundary conditions on the airfoil surface.

Specifies the number of cycles to run.

Specifies the number of iterations per cycle and print frequency to list output file.

Specifies the CFL number.

Basics of Time-Marching Methods

The finite-volume formulation yielded the ordinary differential equation

$$\frac{d\hat{Q}}{dt} = \hat{R}$$

This equation is non-linear in that

$$\hat{Q}(\vec{r},t) \quad \text{and} \quad \hat{R}\left(\hat{Q},\vec{r},t\right)$$

The variable *t* is *time* and indicates that the solution $\hat{Q}$ may be time-varying (unsteady). The time variable gives the equation a mathematically *hyperbolic* character. That is, the solution is dependant on the solutions at previous times. We can use this trait to develop *time-marching numerical methods* in which we start with an initial solution (guess) and march the equations in time while applying boundary conditions. The time-varying solution will evolve or the solution will asymptotically approach the steady-state solution.

Glenn Research Center

Methods for Steady Problems

If we assume the solution is steady, and so, time is not a variable, then the equation takes the form

$$\hat{R}\left(\hat{Q},\vec{r}\right)=0$$

we have at least a few other options for numerical methods:

Iterative Methods. These methods assume the equation is mathematically elliptic and start with an initial solution and iterate to converge to a solution.

Direct Methods. These methods also assume the equation is elliptic, but solve the system of equations in a single process.

Space-Marching Methods. These methods assume that equation can be cast as a mathematically parabolic equation in one of the coordinate directions and marched along that coordinate. (i.e. Supersonic flows in *x*-direction).

Glenn Research Center

Time Discretization

We will first consider time-marching numerical methods. We first need to consider a *finite time step* δt

$$\delta t^n = t^{n+1} - t^n$$

where n is the index for time. The δ indicates a finite step (not differential).

A couple of concepts:

1. The marching of the equation over a time step can occur over one or more *stages* and one or more *iterations*.
2. The marching can be done *explicitly* or *implicitly*. An *explicit method* uses known information to march the solution. An *implicit method* uses known and unknown information and requires solving a local system of equations.

Euler Methods

To demonstrate these concepts, consider the Euler methods, both explicit and implict. The methods are single-stage and first-order accurate in time. The the left-hand side of the equation is discretized as

$$\frac{\delta \hat{Q}^n}{\delta t^n} = \hat{R}$$

where $$\delta \hat{Q}^n = \hat{Q}^{n+1} - \hat{Q}^n$$

and so, $$\hat{Q}^{n+1} = \hat{Q}^n + \delta t^n \hat{R}$$

But how to discretize $\hat{R}$ in in time?

Explicit: $\hat{Q}^{n+1} = \hat{Q}^n + \delta t^n \hat{R}^n$ (Unstable!)

Implicit: $\hat{Q}^{n+1} = \hat{Q}^n + \delta t^n \hat{R}^{n+1}$

Linearization of $\hat{R}^{n+1}$

The Euler Implicit Method introduced an implicit right-hand-side term. This term is approximated using a local linearization of the form

$$\hat{R}^{n+1} = \hat{R}^n + \left(\frac{\partial \hat{R}}{\partial \hat{Q}}\right)^n \delta\hat{Q}^n + \mathrm{O}\left(\delta t^2\right)$$

The flux Jacobian can be defined as $\hat{\mathbf{A}} = \dfrac{\partial \hat{R}}{\partial \hat{Q}}$

such that the Euler implicit method can be expressed as

$$\delta\hat{Q}^n = \delta t\left(\hat{R}^n + \hat{\mathbf{A}}^n\,\delta\hat{Q}^n\right)$$

or

$$\left[\mathbf{I} - \delta t\,\hat{\mathbf{A}}\right]^n \delta\hat{Q}^n = \delta t^n \hat{R}^n$$

where **I** is the identity matrix.

Trapezoidal Time Difference

Trapezoidal (mid-point) differencing for second-order accuracy in time:

$$\frac{d\hat{Q}}{dt} \approx \frac{1}{2}\left[\left(\frac{d\hat{Q}}{dt}\right)^{n} + \left(\frac{d\hat{Q}}{dt}\right)^{n+1}\right]$$

Make substitution of equation,

$$\frac{\delta\hat{Q}^{n}}{\delta t^{n}} = \frac{1}{2}\left[\left(\hat{R}\right)^{n} + \left(\hat{R}\right)^{n+1}\right]$$

Use linearization to form

$$\left[\mathbf{I} - \frac{1}{2}\delta t\,\hat{\mathbf{A}}\right]^{n}\delta\hat{Q}^{n} = \delta t^{n}\hat{R}^{n}$$

Three-Point Backward Time Difference

Three-point backward differencing for second-order accuracy in time:

$$\frac{d\hat{Q}}{dt} \approx \frac{3\hat{Q}^{n+1} - 4\hat{Q}^{n} + \hat{Q}^{n-1}}{2\delta t}$$

This results in the form

$$\left[\mathbf{I} - \frac{2}{3}\delta t\,\hat{\mathbf{A}}\right]^{n} \delta\hat{Q}^{n} = \frac{1}{3}\delta\hat{Q}^{n-1} + \frac{2}{3}\delta t^{n}\hat{R}^{n}$$

Glenn Research Center

MacCormack Method

An example of a multi-stage, explicit method is the MacCormack Method,

Stage 1: $$\hat{Q}^{(1)} = \hat{Q}^{n} + \delta t^{n} \hat{R}^{n}$$

Stage 2: $$\hat{Q}^{(2)} = \hat{Q}^{(1)} + \delta t^{n} \hat{R}^{(1)}$$

and then $$\hat{Q}^{n+1} = \frac{1}{2}\left(\hat{Q}^{n} + \hat{Q}^{(2)}\right)$$

Glenn Research Center

Runge-Kutta Method

Another example of a multi-stage, explicit method is the Runge-Kutta Method. The four stage method has the form:

Stage 1: $\hat{Q}^{(1)} = \hat{Q}^n + \alpha_1 \, \delta t^n \hat{R}^n$

Stage 2: $\hat{Q}^{(2)} = \hat{Q}^n + \alpha_2 \, \delta t^n \hat{R}^{(1)}$

Stage 3: $\hat{Q}^{(3)} = \hat{Q}^n + \alpha_3 \, \delta t^n \hat{R}^{(2)}$

Stage 4: $\hat{Q}^{(4)} = \hat{Q}^n + \alpha_4 \, \delta t^n \hat{R}^{(3)}$

and then $\hat{Q}^{n+1} = \hat{Q}^{(4)}$

Typical coefficients are: $\alpha_1 = 1/4, \ \alpha_2 = 1/3, \ \alpha_3 = 1/2, \ \alpha_4 = 1.$

Time Step Size Control

The size of the time step δt used in the time-marching methods requires several considerations:

- Resolution of time variation (time scale of a fluid particle).
- Stability of the numerical method.
- CFL stability condition:

$$\delta t \le \frac{\nu \, \Delta x}{\hat{\lambda}}$$

ν, Courant-Friedrichs-Lewy (CFL) number (explicit methods generally require $\nu < 1$, but implicit methods allow larger numbers).

Δx, Resolution of the finite-volume cell.

$\hat{\lambda}$, Eigenvalues (wave speeds).

Glenn Research Center

Time Steps, Iterations, and Cycles

The numerical method marches the solution over a *time step* δt. An *iteration* is the numerical process of taking a single time step. A *cycle* is one or more iterations.

Numerical errors exist in most methods that may not result in a second-order solution in time over one iteration. Multiple *sub-iterations* of the numerical method may be needed to remove these errors.

One such method for performing these sub-iterations is the Newton Iterative Method.

Newton Iterative Method

Consider the "standard" Newton iterative method for finding a root $f(x) = 0$,

$$f^{m+1} = f^{m} + f_x^{m}\left(x^{m+1} - x^{m}\right) = 0$$

which can be re-written as

$$x^{m+1} = x^{m} - \frac{f^{m}}{f_x^{m}} \qquad \text{or} \qquad f_x^{m}\,\delta x^{m} = -f^{m}$$

where m is the iteration index. Now consider the equations we wish to solve and substitute them into the above form

$$x \Rightarrow \hat{Q} \qquad f \Rightarrow \hat{Q}_t - \hat{R} = 0 \qquad f_x \Rightarrow -\frac{\partial \hat{R}}{\partial \hat{Q}} = -\hat{\mathbf{A}}$$

Newton Iterative Method (continued)

Substituting these into the Newton iteration equation yields

$$-\hat{\mathbf{A}}^m \delta\hat{Q}^m = -\left(\hat{Q}_t - \hat{R}\right)^m$$

Substituting a three-point, backward time difference for the time derivative, applying some other small approximations, yields.

$$\left[\mathbf{I} - \frac{2}{3}\delta t\,\hat{\mathbf{A}}\right]^m \delta\hat{Q}^m = -\left(\hat{Q}^m - \frac{4}{3}\hat{Q}^n + \frac{1}{3}\hat{Q}^{n-1}\right) + \frac{2}{3}\delta t^n \hat{R}^m$$

The right-hand-side of this equation is simply the discretized form of the equation

$$\hat{Q}_t - \hat{R} = 0$$

which is the equation we wish to solve. Iterating the previous equation until the right-hand side becomes zero will assure that the equation is solved with the errors reduced.

ADI / Approximate Factoring

The Jacobian matrix $\hat{\mathbf{A}}$ is three-dimensional. We must now approximate the Jacobian for the finite-volume cell. We consider a hexahedral cell with generalized coordinates (ξ, η, ζ). We can write the Jacobian as the sum

$$\hat{\mathbf{A}} = \hat{\mathbf{A}}_\xi + \hat{\mathbf{A}}_\eta + \hat{\mathbf{A}}_\varsigma$$

Substituting this into the left-hand side of the Euler Implicit Method results in

$$\left[\mathbf{I} - \delta t\left(\hat{\mathbf{A}}_\xi + \hat{\mathbf{A}}_\eta + \hat{\mathbf{A}}_\xi\right)\right]\delta\hat{Q}$$

This can be "factored" to allow a series a 3 one-dimensional numerical solutions. The factoring neglects third-order terms and higher and results in the form

$$\left[\mathbf{I} - \delta t\,\hat{\mathbf{A}}_\xi\right]\left[\mathbf{I} - \delta t\,\hat{\mathbf{A}}_\eta\right]\left[\mathbf{I} - \delta t\,\hat{\mathbf{A}}_\zeta\right]\delta\hat{Q} = \hat{R}$$

This approach can be applied to the other implicit methods discussed.

ADI / AF (continued)

The equations can then be solved in a series of one-dimensional solutions of the form:

$$\left[\mathbf{I} - \delta t\, \hat{\mathbf{A}}_{\xi}\right]\delta\hat{Q}_1 = \hat{R}$$

$$\left[\mathbf{I} - \delta t\, \hat{\mathbf{A}}_{\eta}\right]\delta\hat{Q}_2 = \delta\hat{Q}_1$$

$$\left[\mathbf{I} - \delta t\, \hat{\mathbf{A}}_{\zeta}\right]\delta\hat{Q}^n = \delta\hat{Q}_2$$

The inviscid part of the Jacobian matrices can undergo further *diagonalization* to form a scalar penta-diagonal system that is numerically easier to invert, and which reduces computational effort. However, this reduces the implicit method to first-order accuracy in time.

Glenn Research Center

Solution Convergence Acceleration

There exists several methods for accelerating the convergence of the solution to a steady-state solution:

- Use a uniform global CFL number which results in varying local time steps. Thus larger time steps are used in regions of larger cells.
- Use an incrementing CFL number that starts with a small CFL number to get past initial transients then increases the CFL number to converge.
- Limit the allowable change in the solution, δQ, over an iteration.
- Locally fix bad solution points by replacing the bad solution point with an average of local points.

Glenn Research Center

Other Methods Not Discussed

Several other solution algorithms are available in WIND that have not been discussed:

- Jacobi Method
- Gauss-Seidel Method
- MacCormack's First-order Modified Approximate Factorization (MAFk)
- ARC3D 3-factor Diagonal Scheme

Glenn Research Center

Summary on Use of Marching Methods

Time-marching algorithms set by use of following keywords:

IMPLICIT
NAVIER-STOKES ITERATIONS
NEWTON
STAGES
DQ LIMITER

Details of keyword usage available in on-line documentation.

Default scheme of Euler Implicit method with Approximate Factorization is robust for steady-state flow simulations.

Space Marching for supersonic streamwise flows use MARCHING keyword.

Flux Numerical Methods

The finite-volume formulation of the conservation equations resulted in the equation

$$\frac{d\hat{Q}}{dt} = \hat{P} - \hat{F}$$

where $\hat{F}$ was the flux of the flow across the control surface resulting from the approximation of the surface integral. For a finite-volume cell, the flux was expressed as

$$\hat{F} = \sum_{f=1}^{nf} \hat{F}_f$$

Where

$$\hat{F}_f = \left[(\vec{v} - \vec{g})Q - \mathbf{D}\right]_f \cdot (\hat{n}\, dS)_f$$

It was assumed that the flux was uniform over the cell face.

Glenn Research Center

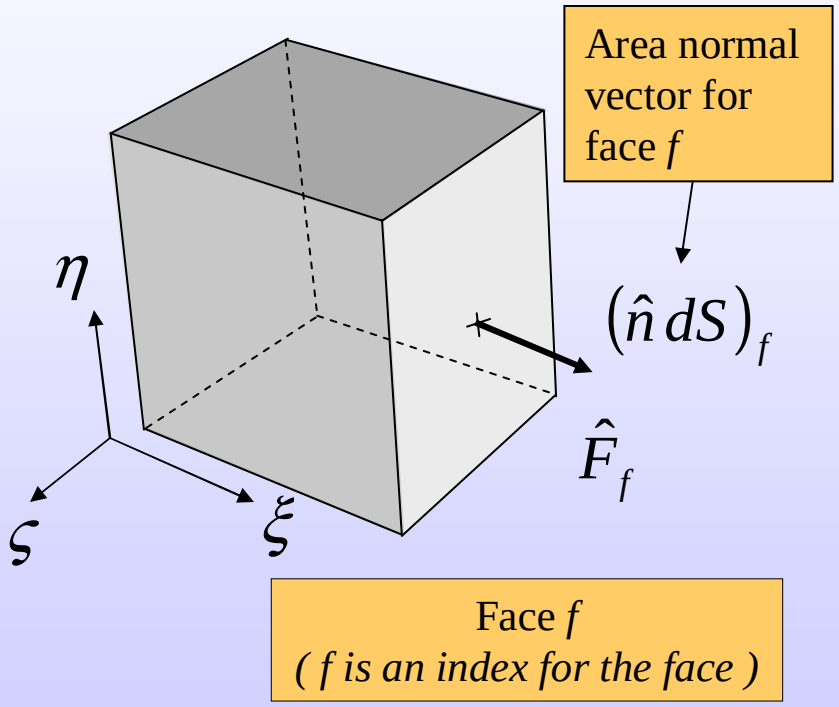
Fluxes on a Hexahedral Cell

A hexahedral cell contains 6 quadrilateral faces, thus

$$\hat{F} = \sum_{f=1}^{nf=6} \hat{F}_f$$

where again,

$$\hat{F}_f = \left[(\vec{v} - \vec{g}) Q - \mathbf{D} \right]_f \cdot (\hat{n}\, dS)_f$$



Numerical Flux on a Cell Face

The numerical flux on a cell face is

$$\hat{F}_f = \left[(\vec{v} - \vec{g})Q - \mathbf{D}\right]_f \cdot (\hat{n}\, dS)_f$$

The normal area vector $(\hat{n}\, dS)_f$ is usually easily defined for a quadrilateral or triangular cell face. The focus of the rest of this discussion is on numerical methods for computing

$$\left[(\vec{v} - \vec{g})Q - \mathbf{D}\right]_f$$

at a cell face.

We first will assume that $\vec{g}$ is a known velocity for the cell face.

Numerical Flux on a Cell Face

We first consider that we have the states of the flow on the "left" and "right" of the cell face, Q_L and Q_R. Our objective is to find the cell face flux.

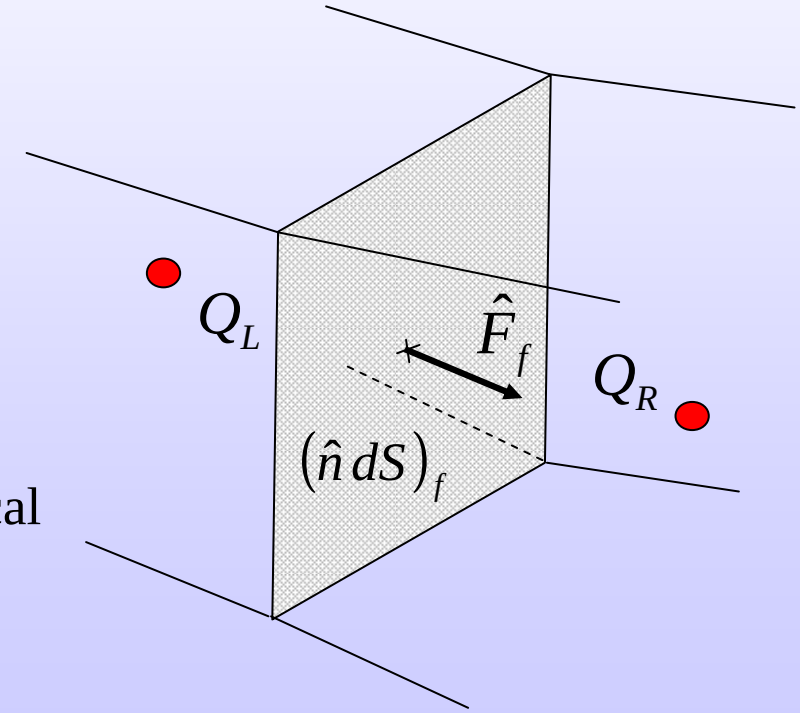
$$\hat{F}_f = \hat{F}_f(Q_L, Q_R)$$

One can define

$$\hat{F}_L = \hat{F}_f(Q_L) \quad \hat{F}_R = \hat{F}_f(Q_R)$$

A consistency condition for the numerical flux is that if $Q_L = Q_R$, then

$$\hat{F}_L = \hat{F}_R$$



Glenn Research Center

Central Difference Method

A central difference method for computing the flux is simply

$$\hat{F}_f = \frac{1}{2}\left(\hat{F}_L + \hat{F}_R\right)$$

The central-difference method works okay for elliptic components of the flux because there is no preferred direction for the propagation of information.

A simple central difference is often unstable, especially in the presence of strong gradients. One solution is to add some second-order and fourth-order dissipation (artificial viscosity) to the flux.

$$\hat{F}_f = \frac{1}{2}\left(\hat{F}_L + \hat{F}_R\right) + D^{(2)} + D^{(4)}$$

Methods for computing $D^{(2)}$ and $D^{(4)}$ vary, but generally use second and fourth-order differences with switches to handle variations in Q.

Glenn Research Center

Use of Central Difference Method

For the Navier-Stokes equations, the viscous shear stress and heat flux terms in the viscous component $\mathbf{D}^V$ of the non-convective component **D** are elliptic and those flux components can be computed with the central difference method.

Similarly, the fluxes of the turbulence and chemistry equations can be computed using the central difference method.

The convective portion of the flux and the pressure term in the inviscid component $\mathbf{D}^I$ of the Navier-Stokes equation have a hyperbolic character. This wave nature can be put to use to compute the flux using *upwind methods*.

Upwind Methods

We expressed the non-convective portion of the flux of the Navier-Stokes equation as

$$\mathbf{D}_{NS} = \mathbf{D}_{NS}^{I} + \mathbf{D}_{NS}^{V}$$

This results in the cell-face flux being expressed as

$$\hat{F}_f = \left[(\vec{v} - \vec{g})Q_{NS} - \mathbf{D}_{NS}^{I} - \mathbf{D}_{NS}^{V}\right]$$

or

$$\hat{F}_f = \hat{F}_f^{\mathbf{I}} + \hat{F}_f^{\mathbf{V}}$$

We will now focus on computing the inviscid flux $\hat{F}_f^{\mathbf{I}}$ using upwind methods. The focus will be on the use of Roe's Upwind Flux-Difference Splitting Method.

Roe Upwind Flux-Difference Method

The Roe upwind flux-difference method computes the inviscid flux as:

$$\hat{F}_f^I = \frac{1}{2}\left(\hat{F}_L^I + \hat{F}_R^I\right) - \frac{1}{2}\Delta\hat{F}_f$$

where $\Delta\hat{F}$ is the flux difference computed as,

$$\Delta\hat{F} = \Delta\hat{F}^+ + \Delta\hat{F}^-$$

$$\Delta\hat{F}^+ = \sum_{m=1}^{5} \lambda_m^{(+)}\, \vec{r}_m\, \delta w_m$$

$$\Delta\hat{F}^- = \sum_{m=1}^{5} \lambda_m^{(-)}\, \vec{r}_m\, \delta w_m$$

Roe's method is the default flux method in WIND.

Glenn Research Center

Roe Upwind Flux-Difference Method

The λ_m are the eigenvalues that represent the speed of the waves. The (+) indicate positive eigenvalues and the (-) indicates negative eigenvalues.

The $\vec{r}_m$ are the right eigenvectors that represent the direction of propagation of the waves.

The δw_m are the Riemann invariants and represent the strength of the wave,

$$\delta w_1 = \delta\rho - \frac{\delta p}{c^2}$$

$$\delta w_2 = n_1\,\delta w - n_3\,\delta u$$

$$\delta w_3 = n_2\,\delta u - n_1\,\delta v$$

$$\delta w_4 = \frac{\delta p}{\rho c} + n_1\,\delta u + n_2\,\delta v + n_3\,\delta w$$

$$\delta w_5 = \frac{\delta p}{\rho c} - n_1\,\delta u + n_2\,\delta v + n_3\,\delta w$$

Roe Upwind Flux-Difference Method

The differentials are computed as

$$\delta\rho = \rho_R - \rho_L$$
$$\delta p = p_R - p_L$$
$$\delta u = u_R - u_L$$
$$\delta v = v_R - v_L$$
$$\delta w = w_R - w_L$$

Flow properties at the face are computed using Roe-averaging

$$\rho^2 = \rho_R \rho_L$$

$$u = \frac{\rho_L^{1/2} u_L + \rho_R^{1/2} u_R}{\rho_L^{1/2} + \rho_R^{1/2}}$$

Similar for computing v, w, and h_t.

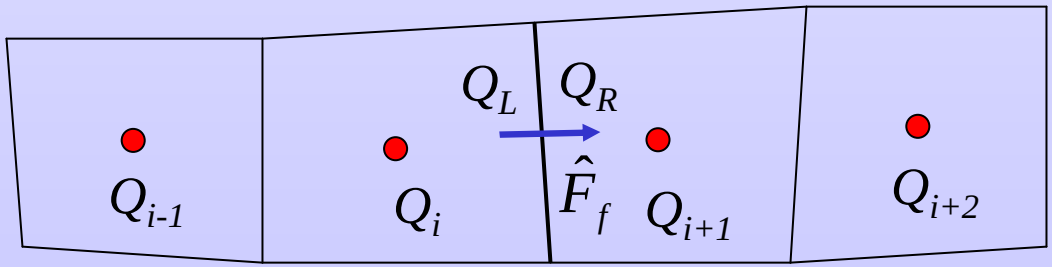
Higher-Order Projection

The choice of values of Q_L and Q_R have several option:

1) Use the values of the finite-volume cells to the "left" and "right" of the face. This is a zero-order evaluation and will result in a *spatially first-order flux*.

2) Use an extrapolation of neighboring finite-volume cells to form a first-order evaluation of Q at the face. This will result in a *spatially second-order flux*.

$$Q_L = Q_i + \frac{1}{2}(Q_i - Q_{i-1}) \qquad Q_R = Q_{i+1} - \frac{1}{2}(Q_{i+2} - Q_{i+1})$$



Variation Limiting

The simple extrapolation formulas assume a smooth variation of Q; however, discontinuities in Q are possible (i.e. shocks). Need some mechanism to sense such discontinuities and limit the variation of Q in these extrapolation formulas. Modify the extrapolations by introducing a limiter ϕ,

$$Q_L = Q_i + \frac{1}{2}\phi(Q_i - Q_{i-1})$$

$$Q_R = Q_{i+1} - \frac{1}{2}\phi(Q_{i+2} - Q_{i+1})$$

This gets into the topic of TVD (Total Variational Diminishing) flux limiting methods, which we will not get into here. The essential role of the limiter is to make $\phi \to 0$ in the presence of large variations, which make the flux spatially first-order.

Examples of Limiters

The possible functions (and theory) for limiters is varied. A couple examples include:

Superbee: $$\phi(r) = \max\left[0, \min\left(2r, 1\right), \min\left(r, 2\right)\right]$$

Chakravarthy: $$\phi(r) = \max\left[0, \min\left\{r, \beta\right\}\right]$$

Where r is some ratio of the flow properties and indicates the amount of variation in the solution. An example is

$$r = \frac{\delta Q_{i+1}}{\delta Q_i}$$

The β is a compression parameter $1 \le \beta \le 2$, where a value toward 1 makes the limiter more dissipative.

Flux Vector Splitting

An alternative to flux-difference splitting is flux-vector splitting that considers that the inviscid flux can be linearly separated

$$\hat{F}_f^I = \hat{F}^+ + \hat{F}^-$$

van Leer's flux-vector splitting has the general form of

$$\hat{F}^{\pm} = f_{\pm}^{mass} \begin{bmatrix} 1 \\ u + n_1\left(-v_n \pm 2c\right)/\gamma \\ v + n_2\left(-v_n \pm 2c\right)/\gamma \\ w + n_3\left(-v_n \pm 2c\right)/\gamma \\ \left(q^2 - v_n^2\right)/2 + \left[(\gamma-1)v_n \pm 2c\right]^2/\left[2\left(\gamma^2-1\right)\right] \end{bmatrix}$$

$$f_{\pm}^{mass} = \pm\frac{\rho c\left(M_n \pm 1\right)^2}{4} \qquad v_n = \vec{v}\cdot\hat{n} \qquad M_n = \frac{v_n}{c}$$

Glenn Research Center

Other RHS Methods

Other methods for the “right-hand-side (RHS)” that will not be discussed:

- Methods available for 3rd to 5th –order spatial accuracy.
- Roe’s method is modified to allow non-uniform grids.
- Roe’s method as used in the OVERFLOW code is available.
- Coakley method is available
- HLLE method is available (similar to Roe’s method)

Glenn Research Center

Summary on Use of Flux Methods

Flux algorithms set by use of following keywords:

- RHS
- BOUNDARY TVD
- TVD
- HLLE
- SMOOTHING

Details of keyword usage available in on-line documentation.

Default scheme of second-order Roe's flux difference splitting is fairly robust.

Use first-order scheme and smoothing during initial iterations to help damp out initial transients.

Glenn Research Center

Specifying Boundary Condition Inputs

The inputs for the boundary conditions are specified through some of the following keywords:

ACTUATOR / SCREEN
ARBITRARY INFLOW
BLEED
COMPRESSOR FACE
COUPLING
DOWNSTREAM MACH
DOWNSTREAM PRESSURE
IMPLICIT BOUNDARY
MASS FLOW
OUTFLOW NON-REFLECTING
PERIODIC
VORTEX GENERATOR
WALL FUNCTION
WALL SLIP
WALL TEMPERATURE

Details of keyword usage available in on-line documentation.

Glenn Research Center

Flowfield Initialization

The time-marching approach requires an initial solution for the entire flowfield

- Uniform flowfield based on conditions listed in the FREESTREAM keyword
- Works fine for external flows and some time for internal flows; however, need to consider the mass imbalance created with the initial flowfield.
- Mimic real operations of an inlet (start up), especially for transonic flow.
- ARBITRARY INFLOW keyword can be used to initialize flowfield in a zone. IJK_RANGE can be used to initialize for a range of grid points.

Glenn Research Center

Damping Initial Transients

Initial flowfield likely contains non-physical conditions, such as non-conserved mass flow that may cause temporary problems for the flowfield

Following keywords provide some capability to get past these initial transients:

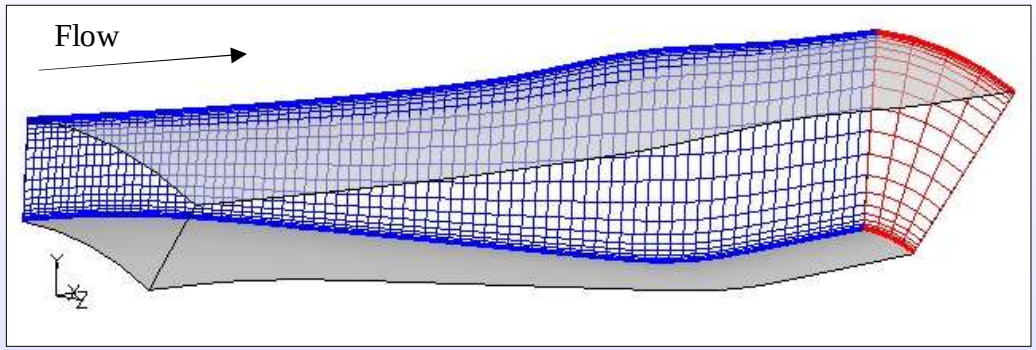
RHS ROE FIRST
TVD FACTOR 0
SMOOTHING
SEQUENCE
DQ LIMITER

There may be significant variation at the start of iterations, but then solution calms down as iterative convergence is approached.

Glenn Research Center

Subsonic Diffuser Initialization 1 of 5

A subsonic diffuser case illustrates some pitfalls of flowfield initialization

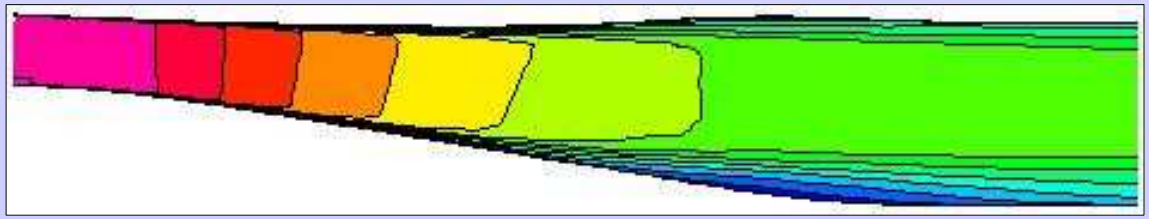


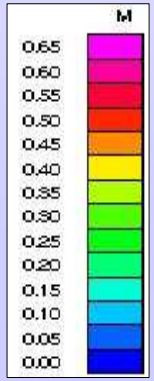
Area ratio, $A_{exit} / A_{inflow} = 1.95$

Inflow: Mach 0.6, $P_t = 10$ psi, $T_t = 520$ R, Area = 7.5 in2

Mass flow through the diffuser is set by the inflow conditions and area

Mass flow = 1.47 lbm/sec



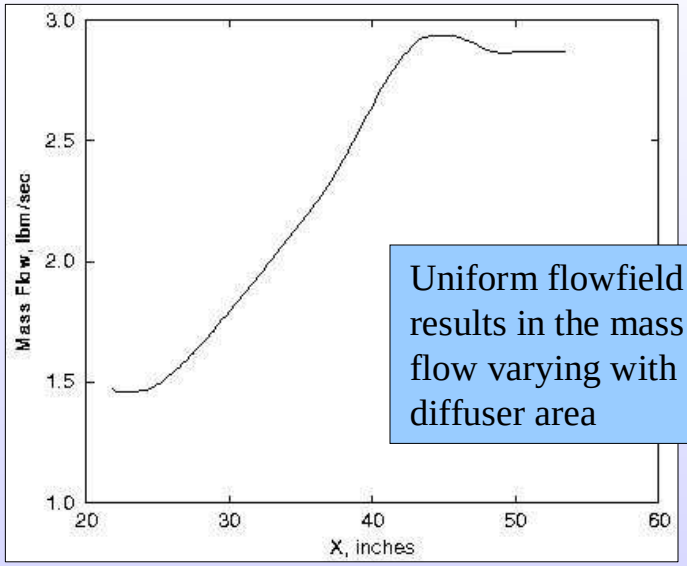


Glenn Research Center

Subsonic Diffuser Initialization 2 of 5

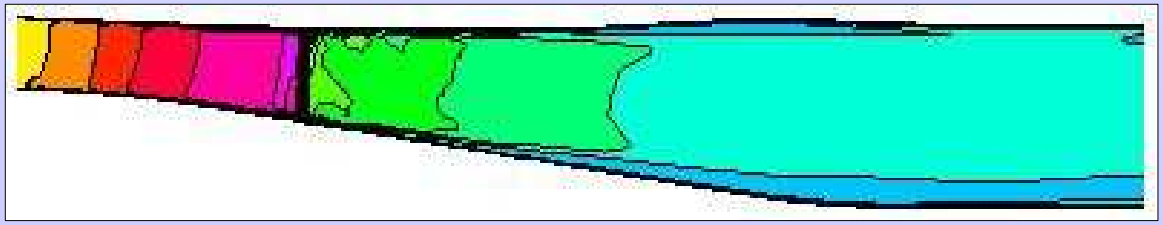
Approach 1: Uniform initial flow based on inflow conditions

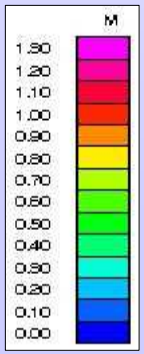
```
freestream total 0.6 10.0 520.0 0. 0.

arbitrary inflow
  total
  hold_totals
  direction along grid lines
  zone 1
  uniform 0.6 10.0 520.0 0. 0.
endinflow

mass flow rate actual 1.47 order one zone 1
```



Uniform flowfield results in the mass flow varying with diffuser area

Initial outflow is 2.87 lbm/sec, but imposing 1.47 lbm/sec at the outflow will result in a hammershock forming in the diffuser that propagates upstream. The shock strength will decrease as it nears the entrance and supersonic flow will be removed. However, the transient existence of the shock may cause problems for the numerical algorithms.





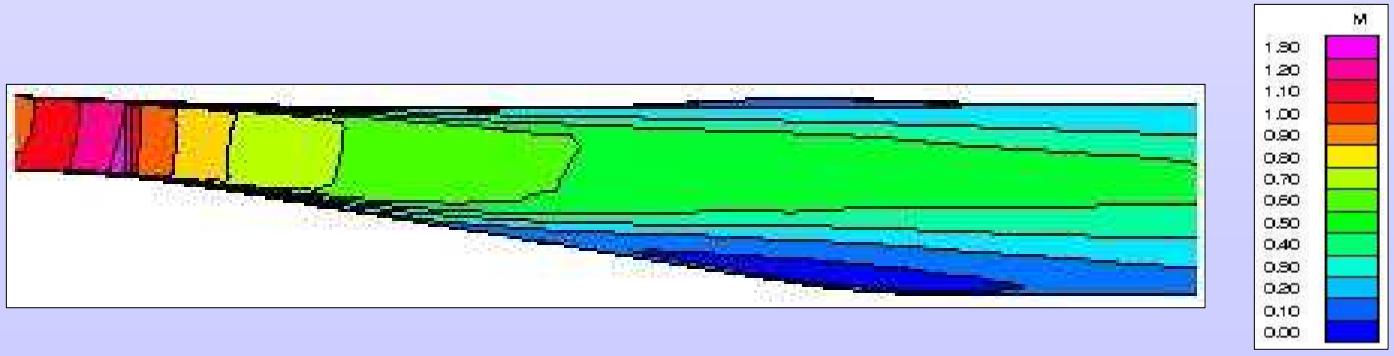
Subsonic Diffuser Initialization 3 of 5

Approach 2: Initialize with a lower Mach number that sets correct outflow

```
freestream total 0.6 10.0 520.0 0. 0.

arbitrary inflow
  total
  hold_totals
  direction along grid lines
  zone 1
  uniform 0.25 10.0 520.0 0. 0.
endinflow

mass flow rate actual 1.47 order one zone 1
```

Set initial Mach number to 0.25 so that outflow would be about 1.47 lbm/sec.

However, accelerations were great enough to result in supersonic flow.



Subsonic Diffuser Initialization 4 of 5

Approach 3: Initialize with a lower Mach number and outflow

```
freestream total 0.6 10.0 520.0 0. 0.

arbitrary inflow
  total
  hold_totals
  direction along grid lines
  zone 1
  uniform 0.1 10.0 520.0 0. 0.
endinflow

/mass flow rate actual 1.0 order one zone 1
/mass flow rate actual 1.2 order one zone 1
mass flow rate actual 1.47 order one zone 1
```

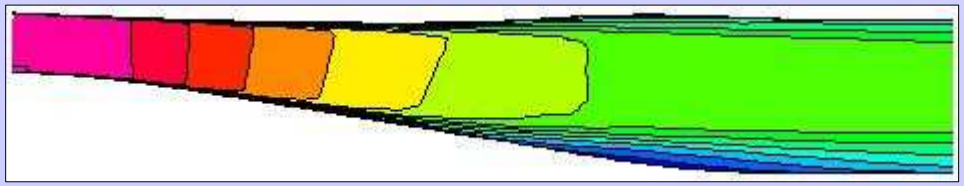
Set the initial Mach number to 0.1 to create a low Mach number flow.

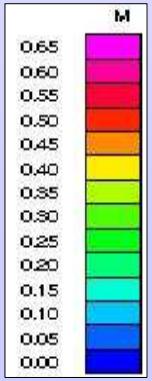
Start with 1.0 lbm/sec outflow for 500 cycles.

Step up to 1.2 lbm/sec outflow for another 500 cycles.

End up with desired 1.47 lbm/sec outflow.

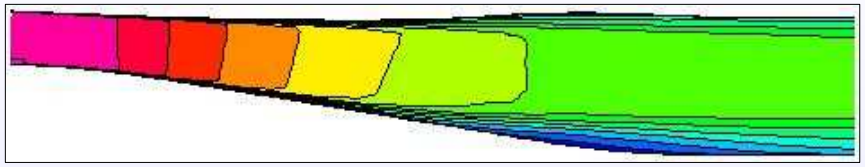
Mimics startup procedures for real inlets

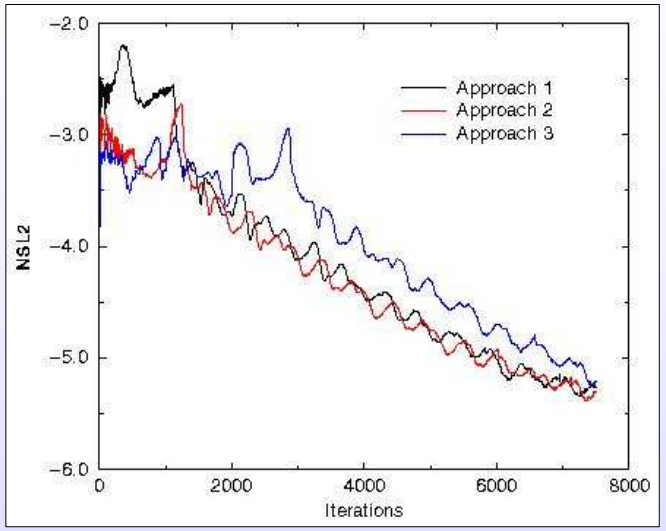




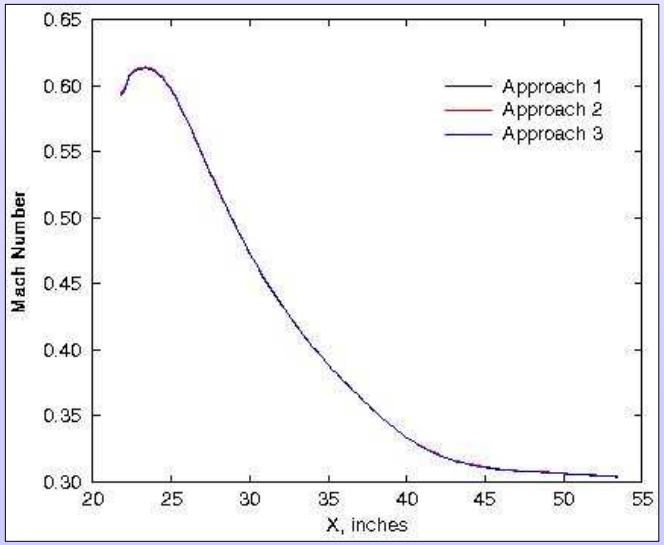
Subsonic Diffuser Initialization 5 of 5

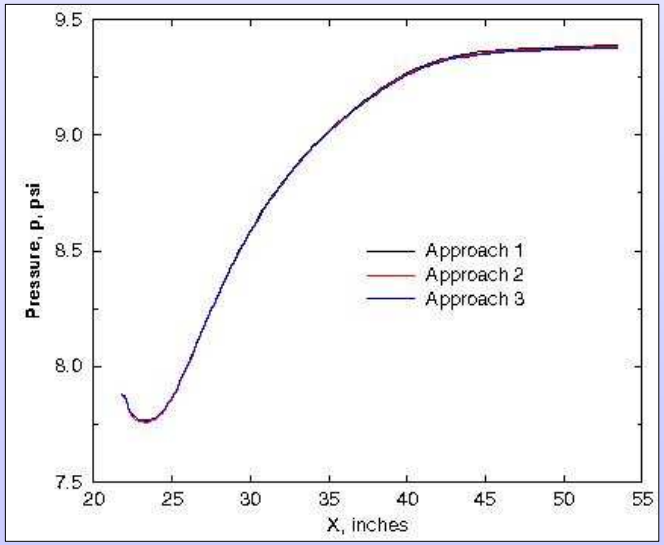
Glenn Research Center





All three approaches yield same solution and have similar convergence behavior, although convergence of Approach 3 lags other approaches. Static pressures and Mach numbers down the diffuser match.





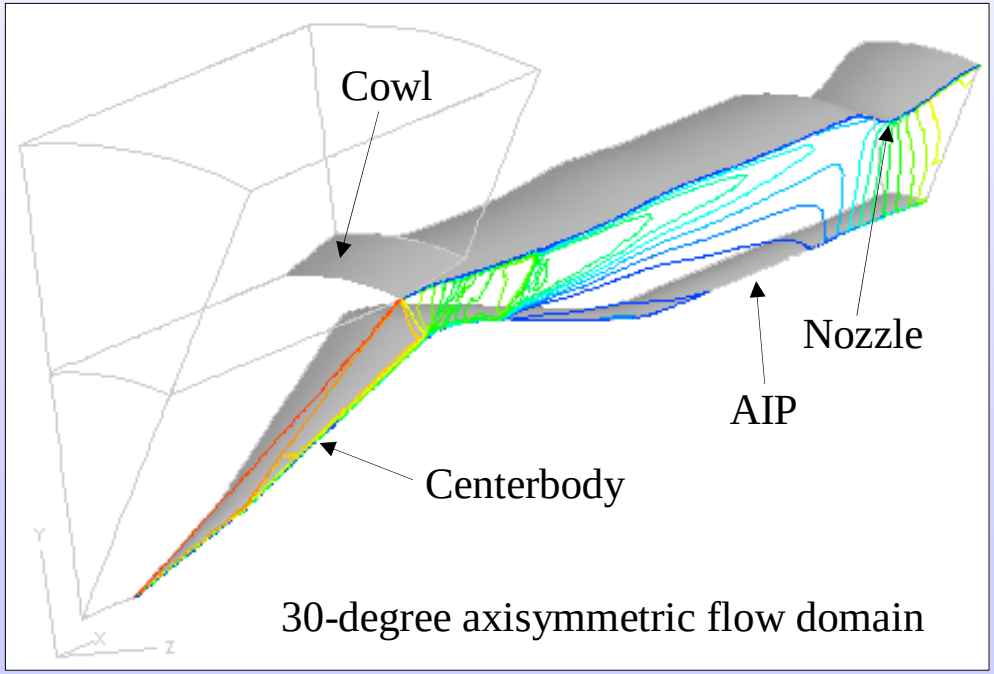
Supersonic Inlet Startup 1 of 2

Mixed-compression supersonic inlet (NASA VDC inlet)

Mach 2.5 freestream

Mach 0.4 outflow at AIP (Aerodynamic Interface Plane)

Nozzle zone for outflow



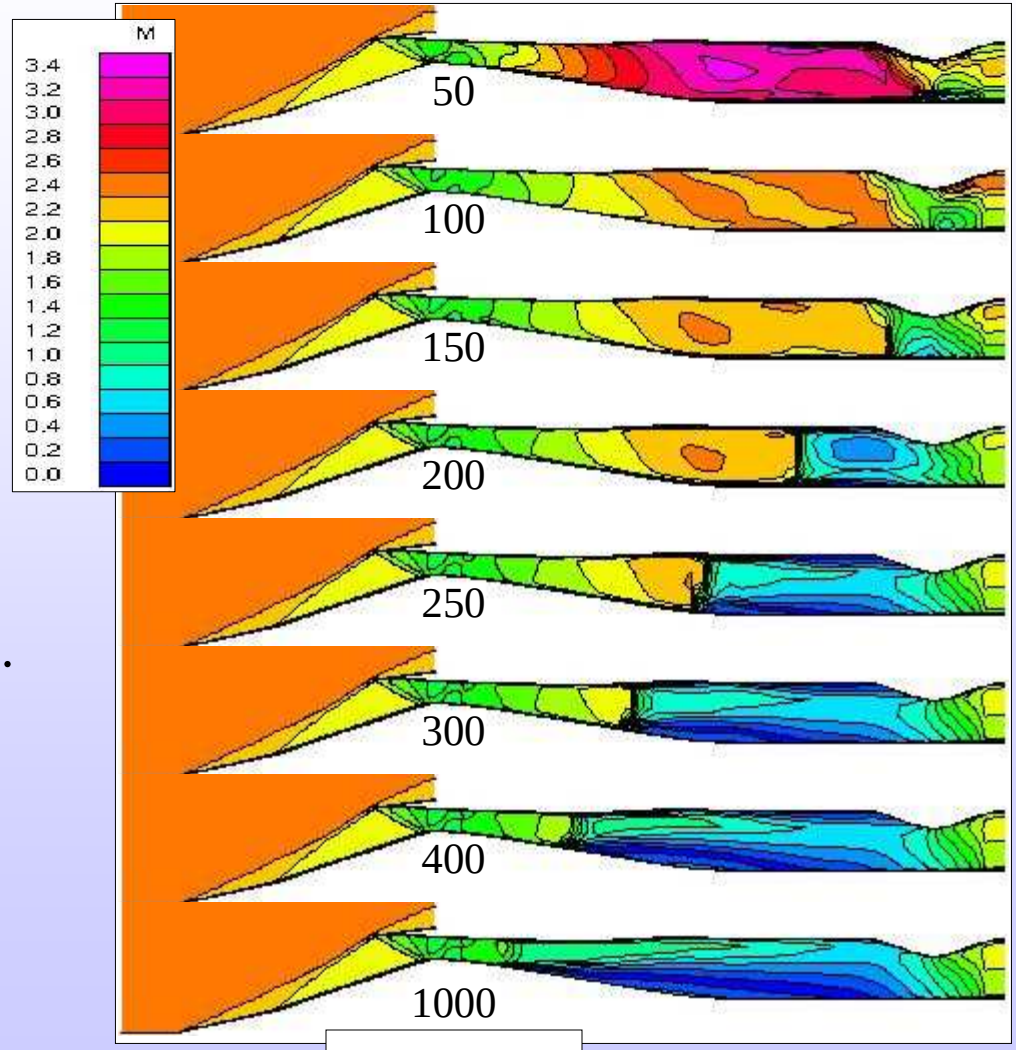
Supersonic Inlet Startup 2 of 2

Glenn Research Center

Initialize with uniform flowfield based on freestream conditions

Forward shock structure sets up rather quickly

Nozzle creates a normal shock that propagates forward in the diffuser until a balance is obtained.



Glenn Research Center

When Things Go Wrong

At times a CFD simulation will fail. Some tips on recovering:

- Check CFL number, using a lower CFL number or increment CFL
- Add more damping with RHS, SMOOTHING, TVD keywords
- Try to "fix" the solution with the FIXER keyword
- Reinitializing the flowfield with ARBITRARY INFLOW and the REINITIALIZE keywords
- Investigate grid problems (excessive skewing or stretching)
- Start with a coarser grid and then refine grid
- Is problem formulated incorrectly?